

**SYSTEM, METHOD, AND COMPUTER PROGRAM PRODUCT FOR
STORING TEST RESULTS IN A DATABASE**

Inventor(s) :

Barnaby Henderson
Half Acre, Cambridge Road
Milton, Cambridge CV4 6AW United Kingdom
Citizen of the United Kingdom

Keir Finlow-Bates
25 the Valley
Comberton, Cambridge CB3 7DF United Kingdom
Citizen of the United Kingdom

Assignee:
Electronic Data Systems Corporation
5400 Legacy Drive, H3-3A-05
Plano, Texas 75024

Matthew S. Anderson
DAVIS MUNCK, P.C.
P.O. Drawer 800889
Dallas, Texas 75380
(972) 628-3600

**SYSTEM, METHOD, AND COMPUTER PROGRAM PRODUCT FOR
STORING TEST RESULTS IN A DATABASE**

TECHNICAL FIELD OF THE INVENTION

5 [0001] The present invention is generally related to
database creation and management.

BACKGROUND OF THE INVENTION

10 [0002] Product testing requires a large number of tests
to be run, and each test result must be stored. For
example, one software product in development can require
that up to 500,000 tests be run each night. It is of great
importance that the developers track and manage regressions
in the product.

15 [0003] If each test result were to be stored in a
separate record in a database, then we would expect in
excess of 150 billion test records (or rows) to be
generated in the test database each year. A database with
this many records has a number of logistical problems,
including search times that are prohibitively high.
20 Further, back-up times and disk space usage are sub-
optimal, the database user experiences slow response, as
portions of the database are paged in and out of memory,
and fixing errors and handling missing records is
difficult. Additional problems arise if the database is to
25 be used over a WAN instead of a LAN.

 [0004] As a result, test departments often "throw away"
test results for previous product releases, and sometimes
even for tests run only a few weeks ago. This can cause

real problems when several different versions of a product are maintained. Similarly, throwing away the history of previous tests is not conducive to the quality process.

5 [0005] There is, therefore, a need in the art for a system, process and computer program product for creating a manageable database containing all the results of the automated test runs, and for other purposes.

SUMMARY OF THE INVENTION

[0006] A preferred embodiment provides a system, method, and computer program product for creating and managing a database. In this embodiment, product test results are efficiently stored in a database in such a way that the database updates test result entries on the basis of result changes as successive build tests are performed.

[0007] The foregoing has outlined rather broadly the features and technical advantages of the present invention so that those skilled in the art may better understand the detailed description of the invention that follows. Additional features and advantages of the invention will be described hereinafter that form the subject of the claims of the invention. Those skilled in the art will appreciate that they may readily use the conception and the specific embodiment disclosed as a basis for modifying or designing other structures for carrying out the same purposes of the present invention. Those skilled in the art will also realize that such equivalent constructions do not depart from the spirit and scope of the invention in its broadest form.

[0008] Before undertaking the DETAILED DESCRIPTION OF THE INVENTION below, it may be advantageous to set forth definitions of certain words or phrases used throughout this patent document: the terms "include" and "comprise," as well as derivatives thereof, mean inclusion without limitation; the term "or" is inclusive, meaning and/or; the phrases "associated with" and "associated therewith," as well as derivatives thereof, may mean to include, be included within, interconnect with, contain, be contained

within, connect to or with, couple to or with, be communicable with, cooperate with, interleave, juxtapose, be proximate to, be bound to or with, have, have a property of, or the like; and the term "controller" means any
5 device, system or part thereof that controls at least one operation, whether such a device is implemented in hardware, firmware, software or some combination of at least two of the same. It should be noted that the functionality associated with any particular controller may
10 be centralized or distributed, whether locally or remotely. Definitions for certain words and phrases are provided throughout this patent document, and those of ordinary skill in the art will understand that such definitions apply in many, if not most, instances to prior as well as
15 future uses of such defined words and phrases.

BRIEF DESCRIPTION OF THE DRAWINGS

[0009] For a more complete understanding of the present invention, and the advantages thereof, reference is now made to the following descriptions taken in conjunction with the accompanying drawings, wherein like numbers designate like objects, and in which:

[0010] Figure 1 depicts a block diagram of a data processing system in which a preferred embodiment can be implemented;

[0011] Figure 2 depicts a flowchart of a process in accordance with a preferred embodiment;

[0012] Figure 3 depicts an exemplary database table created in accordance with an embodiment of the present invention; and

[0013] Figure 4 depicts a pictorial representation of an exemplary database table created in accordance with an embodiment of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

[0014] FIGURES 1 through 4, discussed below, and the various embodiments used to describe the principles of the present invention in this patent document are by way of illustration only and should not be construed in any way to limit the scope of the invention. Those skilled in the art will understand that the principles of the present invention may be implemented in any suitably arranged device. The numerous innovative teachings of the present application will be described with particular reference to the presently preferred embodiment.

[0015] A preferred embodiment provides a system, method, and computer program product for creating and managing a database. In this embodiment, product test results are efficiently stored in a database in such a way that the database updates test result entries on the basis of result changes as successive build tests are performed.

[0016] **Figure 1** depicts a block diagram of a data processing system in which a preferred embodiment can be implemented. The data processing system depicted includes a processor 102 connected to a level two cache/bridge 104, which is connected in turn to a local system bus 106. Local system bus 106 may be, for example, a peripheral component interconnect (PCI) architecture bus. Also connected to local system bus in the depicted example are a main memory 108 and a graphics adapter 110.

[0017] Other peripherals, such as local area network (LAN) / Wide Area Network / Wireless (e.g. WiFi) adapter 112, may also be connected to local system bus 106.

Expansion bus interface 114 connects local system bus 106 to input/output (I/O) bus 116. I/O bus 116 is connected to keyboard/mouse adapter 118, disk controller 120, and I/O adapter 122.

5 [0018] Also connected to I/O bus 116 in the example shown is audio adapter 124, to which speakers (not shown) may be connected for playing sounds. Keyboard/mouse adapter 118 provides a connection for a pointing device (not shown), such as a mouse, trackball, trackpointer, etc.

10 [0019] Those of ordinary skill in the art will appreciate that the hardware depicted in **Figure 1** may vary for particular. For example, other peripheral devices, such as an optical disk drive and the like, also may be used in addition or in place of the hardware depicted. The depicted
15 example is provided for the purpose of explanation only and is not meant to imply architectural limitations with respect to the present invention.

20 [0020] A data processing system in accordance with a preferred embodiment of the present invention includes an operating system employing a graphical user interface. The operating system permits multiple display windows to be presented in the graphical user interface simultaneously, with each display window providing an interface to a different application or to a different instance of the
25 same application. A cursor in the graphical user interface may be manipulated by a user through the pointing device. The position of the cursor may be changed and/or an event, such as clicking a mouse button, generated to actuate a desired response.

5 [0021] One of various commercial operating systems, such as a version of Microsoft Windows™, a product of Microsoft Corporation located in Redmond, Wash. may be employed if suitably modified. The operating system is modified or created in accordance with the present invention as described.

[0022] An exemplary implementation of the method defines the following four fields to exist in the record format:

10 [0023] **Test Name:** A string. Each test must have a unique test name

[0024] **Start Build:** An integer.

[0025] **End Build:** An integer.

15 [0026] **Status:** A string. Typically one of: "pass", "fail", "run time error", or "time out", although other statuses may be relevant to different projects.

20 [0027] Those of skill in the art will recognize that these fields can carry any appropriate label, as long as they are filled and manipulated as described herein. Further, where an action is described herein as dependent on a specific entry in a field, or on an empty field, the skilled artisan will recognize that different implementations can use different values for any of these within the scope of the invention.

25 [0028] Other fields may be added to the record format to allow extra information about individual test results to be stored, for example a comments field to allow test

engineers to add comments for clarification, an action to take field, and a test family name field to allow sensible grouping of batteries of tests.

5 [0029] In the method described below, the master table that holds the results for all the test runs is referred to as the "main table".

[0030] The method processes the test results that, in this example, are produced in the form of ASCII log files generated by the test harnesses as they run the tests.

10 [0031] **Figure 2** depicts a flowchart of a process in accordance with a preferred embodiment.

15 [0032] First, the system will receive the test results from the current build (**step 205**), and the test results are placed in a temporary table (**step 210**). The temporary table is typically stored in the results database, and one record created for each test result.

20 [0033] Each result record in the main table that has an empty "end build" field (of which there should always be one, and only one, per unique test) is matched with the record in the temporary results table that has the same test name (or combination of test name and family name if this is not otherwise unique) (**step 215**).

25 [0034] If a record match is found (**step 220**), then if both records have the same status (**step 225**), the record from the temporary table is ignored (**step 230**).

[0035] If the records have different statuses, the record from the main table has its "end build" field set

equal to the previous build number for which the test was run (**step 245**), and a new record is created in the main table that is a copy of the record in the temporary table (**step 250**). The new record has an empty "end build", and a
5 "start build" set equal to the current build. It also has the same "test name" and "status" as the record in the temporary table.

[0036] If no record match is found in the temporary table then that means that the test was not run. If the
10 "status" of the record in the main table is already "Not Run" then the record from the temporary table is ignored. Otherwise the record from the main table has its "end build" field set equal to the previous build number for which the test was run (**step 235**), and a new record is
15 created in the main table that has an empty "end build", a "start build" set equal to the current build, and a "status" of "Not Run" (**step 240**). It also has the same "test name" as the record from the temporary table.

[0037] The process repeats for the next record in the
20 main table that has an empty "end build" (returning to **step 215**).

[0038] When the process has been completed for all the records in the main table there needs to be a search for new tests that have not been run before. A search is made
25 of the temporary table for records that have a "test name" without a match in the main table. If such records are found then they are added to the main table.

[0039] **Figure 3** depicts a sample database table for four tests that have been run against 100 builds of a software

product. In this example, the current build is 100, with the results stored using the method described in this disclosure.

5 [0040] As can be seen in **Figure 3**, test "PR47001" caused a "Run Time Error" from build 001 to build 099, and a "Fail" on build 100. There is no "end build" entry after build 100 for this test, since the end build entry is only changed when the status/test result changes.

10 [0041] Test PR47002 has a status of "Pass" for all builds after (and including) build 001.

[0042] Test PR47003 caused a "Time Out" from build 001 to build 098, and a "Pass" for all builds after (and including) build 099.

15 [0043] Test PR47004 caused a "Fail" from build 001 to build 099, and a "Pass" for all builds after (and including) build 100.

20 [0044] As can be seen, this test result database is much smaller than a typical test database, in that a new entry is only required when the result for a particular test changes.

25 [0045] **Figure 4** depicts a pictorial representation of the table shown in **Figure 3**, which illustrates why the number of records in the database is reduced. It can be seen that a new record is only required when the result for a particular test changes.

[0046] Each block in the main body of **Figures 3 and 4** represents one database record. Under a traditional "one

record per result" schema, there would be 400 records in the database if the four sample tests shown above were run against 100 builds of the software. With the system described in this disclosure, there are only 7.

5 [0047] In alternate embodiments, the system and method could rely on two or more parameters that could be split into a "start build"/"end build" pair, for example "start time"/"end time", "start position"/"end position", and the record number could be reduced in such cases.

10 [0048] Also, alternately, more than one state change (as recorded in the "status" field) might be measured against; a change in any of these "status"-like fields would trigger the closing of the old record and the creating of a new record as in step 4 part II above.

15 [0049] Further, it is not strictly necessary to record a "start build" for any of the records, as this information can be constructed by calling up all the records with the same test name and ordering them according to their "end build", with the last record being the one with an empty
20 "end build". This method would give the same reduction in the number of records, but would make searching the "history" of a specific test more difficult as extra calculations for the ordering algorithm are required. More importantly it would not record when the test was first
25 run.

[0050] The method described above provides many advantages in a regression detection and management system, including:

[0051] 1. Dramatically reducing the size of the test data stored without losing any information. This reduction in data size results in faster database search and synchronization functionality, and reduces the disk space used in storing the results.

[0052] 2. Providing the test result data in a concise format for our user interface to present to the test engineers and information specialists who work with the results.

[0053] These two advantages, among others, result in users of the disclosed system being able to quickly and efficiently track down and present the regressions and/or changes in the test results.

[0054] Those skilled in the art will recognize that, for simplicity and clarity, the full structure and operation of all data processing systems suitable for use with the present invention is not being depicted or described herein. Instead, only so much of a data processing system as is unique to the present invention or necessary for an understanding of the present invention is depicted and described. The remainder of the construction and operation of data processing system 100 may conform to any of the various current implementations and practices known in the art.

[0055] It is important to note that while the present invention has been described in the context of a fully functional system, those skilled in the art will appreciate that at least portions of the mechanism of the present invention are capable of being distributed in the form of a

instructions contained within a machine usable medium in any of a variety of forms, and that the present invention applies equally regardless of the particular type of instruction or signal bearing medium utilized to actually carry out the distribution. Examples of machine usable mediums include: nonvolatile, hard-coded type mediums such as read only memories (ROMs) or erasable, electrically programmable read only memories (EEPROMs), user-recordable type mediums such as floppy disks, hard disk drives and compact disk read only memories (CD-ROMs) or digital versatile disks (DVDs), and transmission type mediums such as digital and analog communication links.

[0056] Although an exemplary embodiment of the present invention has been described in detail, those skilled in the art will understand that various changes, substitutions, variations, and improvements of the invention disclosed herein may be made without departing from the spirit and scope of the invention in its broadest form.

[0057] None of the description in the present application should be read as implying that any particular element, step, or function is an essential element which must be included in the claim scope: THE SCOPE OF PATENTED SUBJECT MATTER IS DEFINED ONLY BY THE ALLOWED CLAIMS. Moreover, none of these claims are intended to invoke paragraph six of 35 USC §112 unless the exact words "means for" are followed by a participle.